A 2D-3D Model-Based Approach to Real-Time Visual Tracking

Eric Marchand, Patrick Bouthemy, François Chaumette

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

A 2D-3D Model-Based Approach to Real-Time Visual Tracking

Éric Marchand, Patrick Bouthemy and François Chaumette

N°3920

Mars 2000

THÈME 3

ISSN 0249-6399 ISRN INRIA/RR--3920--FR+ENG

INRIA
RENNES

A 2D-3D Model-Based Approach to Real-Time Visual Tracking

Éric Marchand*, Patrick Bouthemy and François Chaumette

Thème 3 — Interaction homme-machine,
images, données, connaissances
Projet Vista

Rapport de recherche n ° 3920 — Mars 2000 — 30 pages

Abstract: We present an original method for tracking, in a monocular image sequence, complex objects which can be approximately modeled by a polyhedral shape. It considers two steps of global transformation, the first one operating in the 2D space and the second one in the 3D space. The first step is able to handle large displacements of the object projection in the image. In involves a 2D motion model estimated by a robust statistical method. Then, we refine the localization of the object silhouette by evaluating the 3D parameters related to the object pose by the iteratively minimizing a nonlinear cost function. It aims at moving the projection of the contours of the object CAD model to the spatial intensity gradients in the image. The proposed tracking method is real-time while being reliable and robust requirements. Real tracking experiments and results embedded in a visual servoing positioning task are reported.

Key-words: Visual tracking, motion estimation, pose computation, visual servoing

(Résumé : tsvp)

* Eric.Marchand@irisa.fr

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

Suivi robuste d'objets en temps-réel : une approche hybride 2D-3D

Résumé : Dans cet article, nous présentons une méthode originale de suivi d'objets complexes approximativement modélisés par une forme polyédrique. Elle enchaîne deux étapes de transformation globale, la première à caractère 2D, la seconde à caractère 3D. Un premier recalage de la silhouette suivie, pouvant appréhender des grands déplacements, est réalisé via l'estimation d'un modèle 2D de mouvement à l'aide d'une méthode robuste. Grâce à cette phase d'initialisation, nous pouvons ensuite évaluer les paramètres d'une transformation 3D par la minimisation itérative d'une fonction de coût non linéaire. Elle consiste à positionner au mieux la projection des arêtes visibles du modèle CAO de l'objet sur les contrastes d'intensité dans l'image. La méthode proposée permet un suivi fiable et robuste en temps réel comme le montrent les résultats présentés.

Mots-clé : Suivi, estimation du mouvement, calcul de pose, asservissement visuel

1 Introduction

We are dealing with the tracking of complex objects in a monocular image sequence. We consider objects which can be approximately modeled by a polyhedral shape. The aim is to use such a tracking method within a robotics context. More precisely, we are concerned with the visual servoing [13] approach that consists in controlling the movements of a robot with respect to image information. This is of key interest for example in a hostile environment as a nuclear power plant. If most of the control issues are now well known and robust control laws can be defined to perform positioning or grasping tasks, the lack of really efficient image processing tools appears as the main shortcoming of these techniques for a wider use. In particular, performing the tracking of the object of interest with respect to the robotics tasks to be achieved is a not too restricted situation, remains an open issue. Indeed, to fulfill visual servoing requirements, image feature extraction must be robust, accurate enough, and computed in real-time (at least at the highest possible rate).


a b c

Figure 1: Tracking features for visual servoing robotic tasks: three levels of complexity (a) tracking artificial landmarks (white dots) for a grasping task (b) tracking specific geometrical features (ellipse and segment) (c) tracking complex objects (a connector) in a textured environment.

Techniques exploited in industrial environments usually involve the tracking of artificial landmarks (Fig. 1.a), or specific well-defined geometrical features (Fig. 1.b) along the image sequences. In order to increase the versatility of visual servoing techniques, such restricted configurations must be alleviated. Our goal is indeed to consider and track complex objects in a non controlled environment (as the connector displayed on Fig. 1.c).

Most of the available tracking techniques can be divided into two main classes: feature-based and model-based. The former approach considers features such as simple geometrical primitives (points, segments [3, 12], circles [20], ...), object contours [1, 14], regions of interest [12], ... The latter explicitly exploits a model of the tracked objects. This model can be a 2D template of the object (active contours [2, 16] or statistical modal representation [6, 17]) or a CAD model [7, 10, 18, 25, 19, 24]. The CAD-based tracking methods aim at registering a reference model with the projection in the image of the object of interest. This usually relies on a pose computation algorithm. This involves the estimation of the 3D rigid transformation that links the object coordinate system to the camera coordinate system, from one view [19], multiple views [5] or from an image sequence [7, 10, 18, 25]. This class of methods usually supplies robust solutions (for example, it can cope with partial occlusion of the objects). Both tracking approaches may use Kalman filters to predict and smoothly estimate the position of the tracked primitives over time.

In our case, we aim at designing a tracking method fulfilling the following properties or constraints: it should be fast and robust, it should require no prior learning step, it should not involve any complex feature extraction (such as contour extraction and linking). Therefore, we have developed an "hybrid" 2D-3D model-based approach that relies both on the estimation of the 2D object motion and of the 3D pose of the object. It supplies a fast and robust tracking of complex objects which can be approximately modeled by a polyhedral shape. More precisely, in a first step, the object image motion between two successive images is represented by a 2D affine motion model. This 2D motion model is estimated, using a robust statistical method, from the computation of the normal displacements along the projected object model contours between two successive images. These normal displacements are determined using the technique described in [4]. The 2D motion model however cannot always account for the real displacement of the object. A second step is required that consists in fitting the projection of the object model on the spatial intensity gradients in the image. This is achieved using an iterative minimization of a non linear energy function with respect to the 3D pose parameters. The main advantages of this two-step method can be outlined as follows. The 2D motion estimation stage allows us to handle large displacements of the object.

It also avoids a prediction step. Indeed, the latter often remains a questionable issue concerning the choice of the evolution model and the detection of model changes. The result of this first stage is exploited to supply an appropriate initialization to the pose estimation. Our model-based tracking only requires a coarse calibration of the camera and a rough model of the object. Both 2D motion estimation and 3D pose estimation do not involve edge detection and image contour extraction. We only process gray level arrays. Both stages are robust to partial occlusions of the object. Finally, real-time tracking is reachable.

The paper is organized as follows. Section 2 describes the 2D motion-based tracking stage that acts as an initialization to the 3D model-based tracking presented in Section 3. Experimental tracking results and real-time visual servoing tasks are reported in Section 4. Section 5 contains concluding remarks.

2 2D motion-based tracking

We first consider that the global transformation between two successive projections of the tracked object in the image plane can be represented by a 2D affine motion model. The goal of this first step is to estimate the parameters of this 2D transformation even in presence of large 2D displacements of the object image. Contrary to usual Kalman filter methods, this motion-based method does not involve any prediction scheme. This means that it does not require in particular the introduction of a state model evolution (e.g., a constant velocity model), and the initialization of the noise variance of the state and measurement models, which are often critical matters.

2.1 Affine and quadratic transformation models.

Let $\mathcal{X}^t = [X_1^t, \ldots, X_n^t]^T$ be a vector formed by the image coordinates X_i^t of points along the projection of the boundaries of the polyhedral object model at time t. This vector is a way captures the shape and position of the image of the tracked object. The object image shape $\mathcal{X}^{t+1}$ at time $t+1$ will be given by:

$$\mathcal{X}^{t+1} = \Psi_\Theta(\mathcal{X}^t) \tag{1}$$

where Ψ_Θ is a 2D affine transformation expressed as:

$$\begin{bmatrix} x_i^{t+1} \\ y_i^{t+1} \end{bmatrix} = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix} \begin{bmatrix} x_i^t \\ y_i^t \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix} = \mathbf{W}(X_i^t)\Theta \tag{2}$$

with $\Theta = (a_1, a_2, a_3, a_4, T_x, T_y)^T$, $X_i^t = (x_i^t, y_i^t)^T$, $X_i^{t+1} = \Psi_\Theta(X_i^t)$, and

$$\mathbf{W}(X) = \begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \end{bmatrix}$$

This transformation is linear wrt. Θ. Displacement vector $\mathbf{d}_i(X_i) = X_i^{t+1} - X_i^t$ can be written as follows:

$$\mathbf{d}_i(X_i) = \mathbf{W}(X_i)\Theta' \tag{3}$$

where $\Theta' = \Theta - (1, 0, 0, 1, 0, 0)^T$.

We have considered above a six-parameters affine model. However, in the case of a planar rigid object, a height-parameters quadratic model exactly accounts for the real 3D transformation. If required, we can alternatively we can also estimate the polynomial 2D motion model represented by $\Theta = (a_0, a_1, a_2, a_3, a_4, a_5, a_6, a_7)$ and given by:

$$\begin{bmatrix} x_i^{t+1} \\ y_i^{t+1} \end{bmatrix} = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} + \begin{bmatrix} a_2 & a_3 \\ a_4 & a_5 \end{bmatrix} \begin{bmatrix} x_i^t \\ y_i^t \end{bmatrix} + \begin{bmatrix} a_6 & a_7 & 0 \\ 0 & a_6 & a_7 \end{bmatrix} \begin{bmatrix} x_i^{t2} \\ x_i^t y_i^t \\ y_i^{t2} \end{bmatrix} = \mathbf{W}(X)\Theta \tag{4}$$

with

$$\mathbf{W}(X) = \begin{bmatrix} 1 & x & y & 0 & 0 & 0 & x^2 & xy \\ 0 & 0 & 0 & 1 & x & y & xy & y^2 \end{bmatrix}. \tag{5}$$

The part of the tracking algorithm concerned with the estimation of the 2D affine motion parameters is articulated into two sub-steps:

- the first one computes normal displacements between two successive images along the projection of the object model contours using the so-called Moving Edges (ME) algorithm [4] ;

- the second one utilizes this normal displacement field to estimate $\widehat{\Theta}'$ by adapting [23] the robust multiresolution estimation technique introduced in.

We now describe these two sub-steps.

2.2 Computing normal displacements.

One of the advantages of the ME method is that it does not require any prior edge extraction. We only manipulate point coordinates and image intensities. Nevertheless, for convenience, we will still use the word "contour" to refer to the list of tracked points. The ME algorithm can be implemented with convolution efficiency, and can lead to real-time computation [3, 4]. We consider a list L^t of pixels along the contour of the projection of the object model a time t. The model fitting step between the 3D object model and the projection of the object of interest in the very first image is performed in a semi-automatic mode as described later in section 4.1. The normal displacement computation process consists in seeking in image I^{t+1} the "correspondent" P_i^{t+1} in the next image of each point $P_i^t \in L^t$ In the direction normal to the contour. We determine a 1D search interval $\{Q_i^j, j \in [-J, J]\}$ in the direction δ of the normal to the contour (see Fig. 2). For each point P_i^t of the list L^t, and for every entire position Q_i^j lying in fact for computational issue in the direction δ^*, closest to δ, from the set $\{0^o, 45^o, 90^o, 135^o\}$), we compute a criterion corresponding to the square root of a log-likelihood ratio ζ^j. The latter is nothing but the absolute sum of the convolution values, computed at P_i^t and Q_i^j respectively in images I^t and I^{t+1}, using a pre-determined mask M_δ function of the orientation of the contour [4].

The new position P_i^{t+1} is given by:

$$Q_i^{j^*} = \arg \max_{j \in [-J,J]} \zeta^j$$

with

$$\zeta^j = \mid I^t_{\nu(P_i)} * M_\delta + I^{t+1}_{\nu(Q_i^j)} * M_\delta \mid$$

providing that ζ^{j^*} is greater than a given threshold λ. $\nu(.)$ is the neighborhood of the considered pixel. Then, pixel P_i^{t+1} given by $Q_i^{j^*}$ is stored in L^{t+1}.

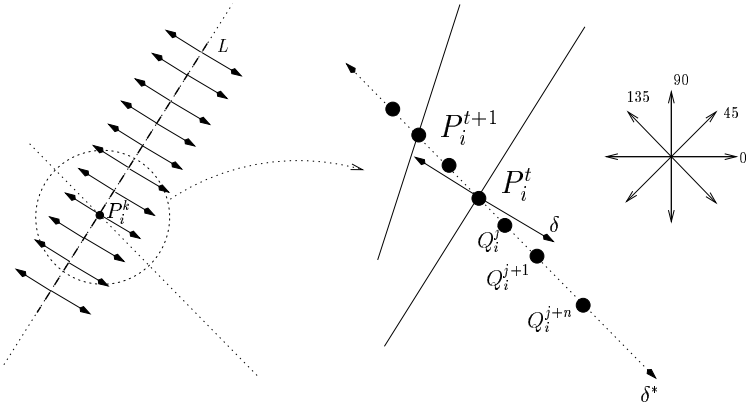

Figure 2: Determining point positions of the tracked object contours s in the next image using the ME algorithm

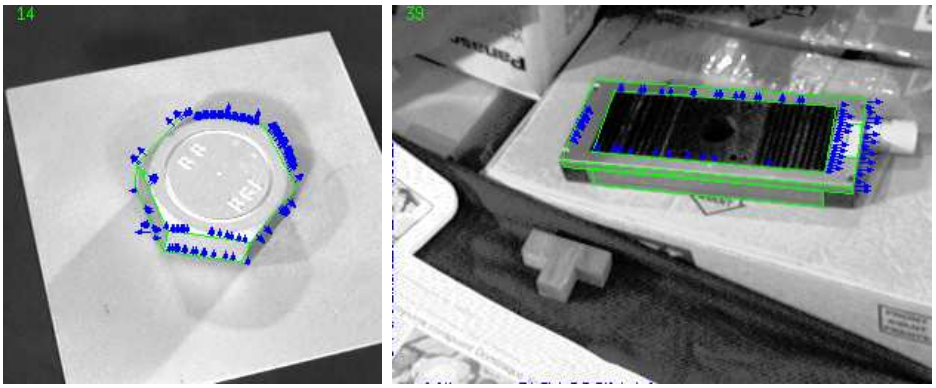

Figure 3: Computed normal displacement vectors on two different examples ($J = \pm 5$, $\lambda = 1500$)

At this step, we have a list of k pixels as well as their displacement component quasi-orthogonal to the object model contour: $(P_i^t, d_i^{\perp *})_{i=1...k}$ (see Fig. 3). This is performed for each new frame, it never requires the extraction of new contours. Since it is a local approach, the presence of partial occlusions of the object only lead to lost some local measurements without perturbing the available ones.

2.3 2D affine transformation estimation.

From $(P_i^t, d_i^{\perp *})_{i=1...k}$, we can estimate the 2D affine transformation Θ'. Using equation (3), we have:

$$d_i^{\perp *} = \mathbf{n}_i^{*T} d(P_i) = \mathbf{n}_i^{*T} \mathbf{W}(P_i)\Theta' \tag{6}$$

where n_i^* is the unit vector quasi-orthogonal to the object model contour at point P_i. Relying on (6), we can use a robust estimator (a M-estimator ρ) to obtain $\widehat{\Theta}'$ as follows [22]:

$$\widehat{\Theta}' = \arg\min_{\Theta'} \sum_{i=1}^{n} \rho\left(d_i^{\perp *} - \mathbf{n}_i^{*T} \mathbf{W}(P_i)\Theta'\right) \tag{7}$$

This robust statistical criterion enables not to be affected by locally incorrect or missing measurements due to shadows, local miss-matching, partial occlusions, etc.

3 3D model-based tracking

3.1 Overview

Knowing the position $\mathcal{X}^t$ of the projection of the contours of the tracked object at time t and the estimated parameters $\widehat{\Theta}'$ of the 2D global affine motion model between t and $t+1$, we are able to compute the positions of points $\mathcal{X}^{t+1}$ at time $t+1$ according to:

$$\mathcal{X}^{t+1} = \Psi_{\widehat{\Theta}}(\mathcal{X}^t)$$

with

$$\widehat{\Theta} = \widehat{\Theta}' + (1, 0, 0, 1, 0, 0)^T$$

However, the 2D transformation cannot generally completely account for the real transformation undergone by the projection of the object (e.g., due to perspective effects, important rotations, non shallow environment), and after a few images tracking may fail. To alleviate this problem, different approaches can be proposed:

- in the case of a planar object, the quadratic 2D transformation fully described the 2D object motion. To avoid the integration of errors over time, the previous 2D tracking process could be completed with a fine registration of the 2D template on the intensity gradients of the images wrt. 2D quadratic motion parameters. Nevertheless, in case of non-planar objects, the introduction of a quadratic motion model does not bring much improvement.
- in a first version of the algorithm, the 2D affine displacement model was augmented with 2D local deformations [11, 23]. However, when adding local deformations, we cannot ensure global 3D rigidity constraints. Moreover, this was highly time consuming.
- finally, we can exploit a rough CAD polyhedral model of the object. This is the approach we present now.

3.2 Fitting CAD model on the spatial intensity gradients.

Our goal is now to fit a CAD model of the tracked object on the intensity spatial gradients in the successive images. To achieve this goal, we consider a pose computation algorithm. Indeed, we have to find the 3D rotation and the 3D translation (i.e., the pose denoted by Φ) that map the object coordinate system with the camera coordinate system.

3.2.1 Initial pose computation

We can use as an appropriate initialization for pose computation the output of the 2D tracking stage. We then have to estimate the pose of the object wrt.

the camera from the positions $\mathcal{X}^{t+1}$ obtained after the first 2D tracking stage described in Section 2. A number of methods to compute pose from points have been proposed. We have used the method designed by Dementhon and Davis [8] completed by Lowe's non-linear method [19]. Dementhon's method calculates the rigid transformation in an iterative way from the knowledge of the coordinates of at least four points in the object coordinate system, and of their corresponding projections in the image. In our case, this information is provided by the new position of points $\mathcal{X}^t$ induced by the estimated 2D transformation: $\Psi_{\widehat{\Theta}}(\mathcal{X}^t)$. Its principle consists in approximating perspective projection by scaled orthographic projection, and then in iteratively modifying the scaled orthographic projection to converge to the perspective projection. We then apply the method proposed by Lowe to improve the pose estimation: Lowe's approach is based on an iterative minimization of a residual using the non linear Levenberg-Marquardt minimization technique. We then get a first estimate of the pose parameters Φ_{init}^{t+1}:

$$\Phi_{init}^{t+1} = f\left(\mathcal{X}^{t+1}\right) = f\left(\Psi_{\widehat{\Theta}}(\mathcal{X}^t)\right)$$

where $f(.)$ denote the pose computation process. Once the pose parameters are available, we can easily determine visible and invisible faces of the object, which is of particular interest for the fitting stage describes in the next paragraph.

However, as this initial pose Φ_{init}^{t+1} is directly related to the 2D tracking process, it has to be still updated to correspond as well as possible to the real new aspect of the object.

3.2.2 Fitting CAD model.

The next step consists in fitting the projection of the object model on the spatial intensity gradients in the image. This is achieved using an iterative minimization wrt. the pose parameters Φ of a non linear cost function using Φ_{init}^{t+1} as initialization. Final pose parameters $\widehat{\Phi}$ are then given by:

$$\widehat{\Phi} = \arg\min_{\Phi} E(\Phi) \tag{8}$$

where the cost function $E(\Phi)$ is defined as:

$$E(\Phi) = -\int_{\Gamma_\Phi} \|\nabla I_{\pi_\Phi(s)}\| ds \tag{9}$$

where Γ_Φ represents the visible part of the 3D object model contours for the pose Φ, and $\nabla I_{\pi_\Phi(s)}$ denotes the spatial gradient of the intensity function at image point $\pi_\Phi(s)$ along $\pi_\Phi(\Gamma_\Phi)$ where π_Φ is the perspective projection function.

The cost defined in equation (9) is the most direct possible one to express the fitting problem. However, we may exploit other available information than the norm of the spatial image gradient. Indeed, when projecting the object model for a given pose Φ, we are able to compute the expected direction of the projected model contour at a 2D point $\varsigma = \pi_\Phi(s)$. If we denote $\mathbf{n}_\Phi(s)$ the unit vector corresponding to this expected direction, the dot product $\nabla I_\varsigma . \mathbf{n}_\Phi(s)$ should be equal ideally to zero, in practice close to zero. Another expression of the cost function exploiting this information can then be considered as follows:

$$E(\Phi) = \int_{\Gamma_\Phi} \frac{\mid \nabla I_\varsigma . \mathbf{n}_\Phi(s) \mid}{\|\nabla I_\varsigma\|^2} ds \tag{10}$$

where we only consider points ς where $\|\nabla I_\varsigma\| > \varepsilon$. This cost expression gives better results in case of textured environments.

The projection function π_Φ depends on the camera intrinsic parameters $\mathcal{I}$. The minimization of the cost function (8) requires that the camera calibration is available. Nevertheless, a rough knowledge of the camera parameters is sufficient. If calibration is wrong, the resulting estimation of Φ will be obviously biased, but the projection of the CAD model in the image, which is the useful information for image tracking purpose, is still correct. On the other hand, these intrinsic camera parameters could also be estimated (or at least updated) on-line. In that case, the criterion to be minimized can be rewritten as follows:

$$\left(\widehat{\Phi}, \widehat{\mathcal{I}}\right) = \arg\min_{(\Phi,\mathcal{I})} \{E(\Phi, \mathcal{I})\} \tag{11}$$

In the general case, we have 11 parameters to estimate (if we consider the radial distortion). In practice, we have only performed experiments dealing with the on-line estimation of the radial distortion.

3.3 Computational issues

To ensure a fast tracking, some computational issues have to be considered especially dealing with the discretization of equations (9) and (10) and with the optimization of criterion (8).

Discretization. Discretization of Γ_Φ can be considered in different ways. If we consider N_e visible model contours to be discretized into N_p 2D points, expression (9) can be rewritten as:

$$E(\Phi) = -\frac{1}{N_e}\sum_{e=1}^{N_e}\left(\frac{1}{N_p^e}\sum_{p=1}^{N_p^e}\|\nabla I_{\varsigma_p^e}\|\right) \tag{12}$$

We have now to determine the $N_e \times N_p^e$ 2D points ς. The first approach is to discretize each contour into N_p^e points s in the 3D space, and then, to project these points in the image plane ($\varsigma = \pi_\Phi(s)$). ς_p^e is thus computed as:

$$\varsigma_p^e = \pi_\Phi\left(s_0^e + \frac{p-1}{N_p^e}\left(s_1^e - s_0^e\right)\right),\ p = 1\ldots N_p^e \tag{13}$$

where s_0^e and s_1^e are the two 3D extremities of the model contour e. A second approach can be considered. The discretization can be performed after the projection of the extremities of the 3D visible object contour in the image. Indeed, as the considered objects are polyhedral, the projection of their contours are also segments and ς_p^e can be computed as:

$$\begin{aligned}\varsigma_p^e &= \pi_\Phi(s_0^e) + \frac{p-1}{N_p^e}\left(\pi_\Phi(s_1^e) - \pi_\Phi(s_0^e)\right),\ p = 1\ldots N_p^e \\ &= \varsigma_0^e + \frac{p-1}{N_p^e}\left(\varsigma_1^e - \varsigma_0^e\right)\end{aligned}$$

These discretizations scheme does not include the distortion term, but knowing K_d, the correct position of the image points can be easily computed. These two versions are similar in term of complexity when distortion is important, but the second one avoids a discretization step in 3D. However, when there is no need to consider radial distortion, the latter approach is indeed more efficient,

since we compute only two projections in the image per segment while the former implies N_p^e projections. Furthermore, there exists efficient algorithms to determine all the pixels (entire positions) attached to a given real segment (e.g., using Bresenham algorithm).

Optimization algorithm. Another important issue is the optimization procedure. Indeed, expressions (9) and (10) are non linear, and involve numerous local minima. To solve this issue we resort to an explicit discrete search algorithm. Generalized Hough transform, consisting in building a cumulative histogram in the pose space, presents two main shortcomings which are the size of the pose space ($\mathbb{R}^6$) and the presence of false peaks. We have rather considered a recursive search algorithm. The method is inspired from a classical algorithm for fast block matching [15]. First, $E(\phi)$ is minimized considering large variation steps of parameters values. When the current minimum is found, the process is iterated with smaller variation steps around this value (see Fig. 4, D is the initial estimate of the camera location, O_1 corresponds to the first minimum found with a first variation step Δ_1, and O_2 is an improved minimum computed from O_1 with a second variation step $\Delta_2 < \Delta_1$.) In practice, the initial solution Φ_{init}^{t+1} is a proper initialization of this search algorithm. Therefore, we can bound the search space. This allows the algorithm to converge very quickly toward an appropriate minimum.

4 Experimental results

Experiments reported hereafter involve various objects. The object complexity is representative of the applications EDF (Électricité de France) is interested in: disassembly and monitoring tasks in the nuclear power plant context. These systems are of interest for the Research and Development division of EdF to achieve maintenance and monitoring tasks in hostile environment. We have selected a nut, a micro-controller testbed, and a serial connector. For all the experiments reported in this section, camera calibration is not precisely known. All the images has been acquired on the IRISA's robotic testbed displayed on Fig. 11.

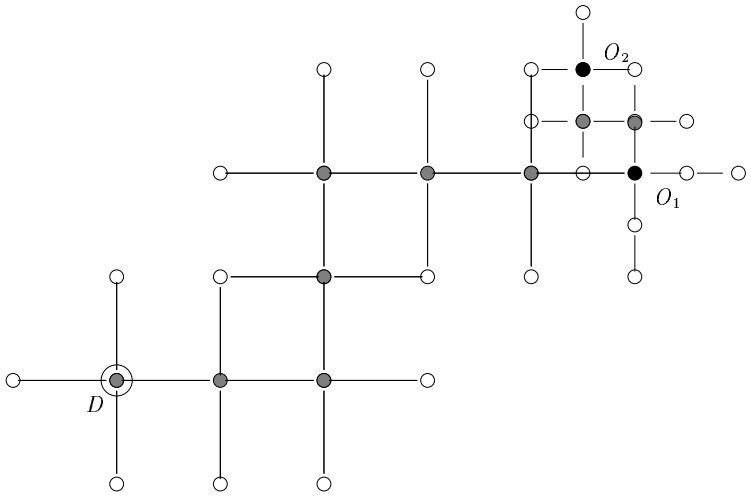

Figure 4: minimization algorithm illustrated for two pose parameters relying on a discrete hierarchical search procedure. In fact, we have six parameters to estimate.

4.1 Initialization of the tracking in the very first image

In the current version of the system described in this paper, initialization of the tracking in the very first image of the sequence is performed partly manually. This means that the user has to click at least four points on both the initial image and the CAD model of the object (see Fig. 5). This is achieved within an interactive procedure ensuring also the matching between the selected model points (vertices of the CAD model) and their corresponding projections in the images located by the user. A completely automatic object localization procedure could be implemented, but this is outside the tracking issue considered in this paper. Let us note that, if the user clicks a minimum of six points, a full camera calibration can be performed (the calibration will be actually rough with this small number of points). The obtained intrinsic parameters can be used afterwards in the pose computation algorithm. After this interactive step a first pose is computed by Dementhon algorithm (Fig. 5.b) which pose is then refined using our CAD fitting stage (Fig. 5.c) to account for imprecision in the manual initialization.

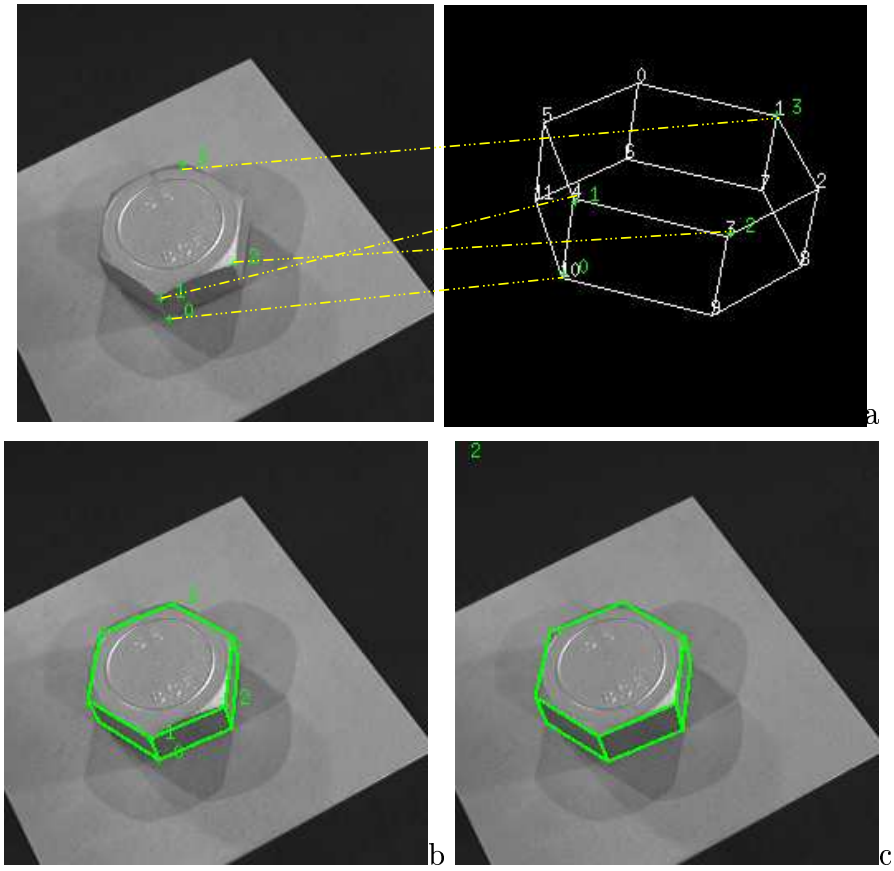

Figure 5: Tracking initialization: (a) selected points in the first image and the CAD model of the object (the virtual doted lines account for the manual 2D-3D registration of the few selected points) (b) back projection of the 3D model using the pose computation process by Dementhon-Davis algorithm (c) back projection of the 3D model after pose computation using our CAD fitting stage.

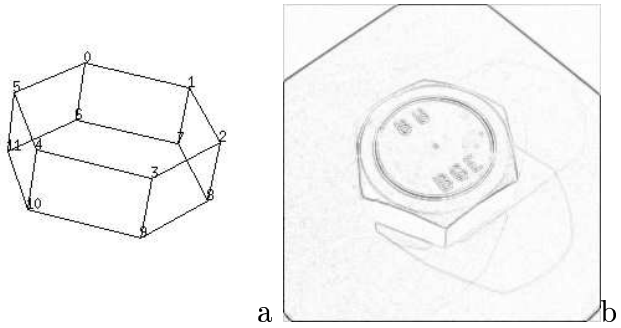

Figure 6: Object of interest: the nut, (a) considered approximate CAD model of the nut, (b) magnitude of spatial intensity gradients in a typical image of the sequence

4.2 Tracking experiments

Nut tracking We first consider the tracking of the nut silhouette in image sequences acquired along predefined trajectories of the camera. Let us point out that we have to deal with low intensity contrast (as it can be seen in the intensity gradient image of Fig. 6.b), with presence of cast shadows, of specularities, ... Moreover, the nut is not exactly polyhedral, since it presents no physically angular ridges. Finally the CAD model has been hand-made which implies low precision. Despite these difficulties, the proposed method has proven its ability to efficiently track this object along long image sequences.

Fig. 7 contains results of the tracking of the nut along a sequence of 44 images. Fig. 7.a shows the results of the tracking if we consider only the 2D motion estimation step. In that case, tracking is performed at video rate (25Hz). However, after a few images, the algorithm is no longer able to track accurately the object shape. The failure is mainly due to the fact that a 2D affine motion model cannot completely account for the projection of the 3D motion of the object. Fig. 7.b reports the results of the tracking involving both stages, 2D motion estimation and 3D pose computation. In that case, tracking can be performed at 10 Hz on a PC 400 Mhz under Linux OS.

We have also validated the performance of the tracking algorithm in the presence of various difficulties. In the experiment of Fig. 8.a, a camera motion

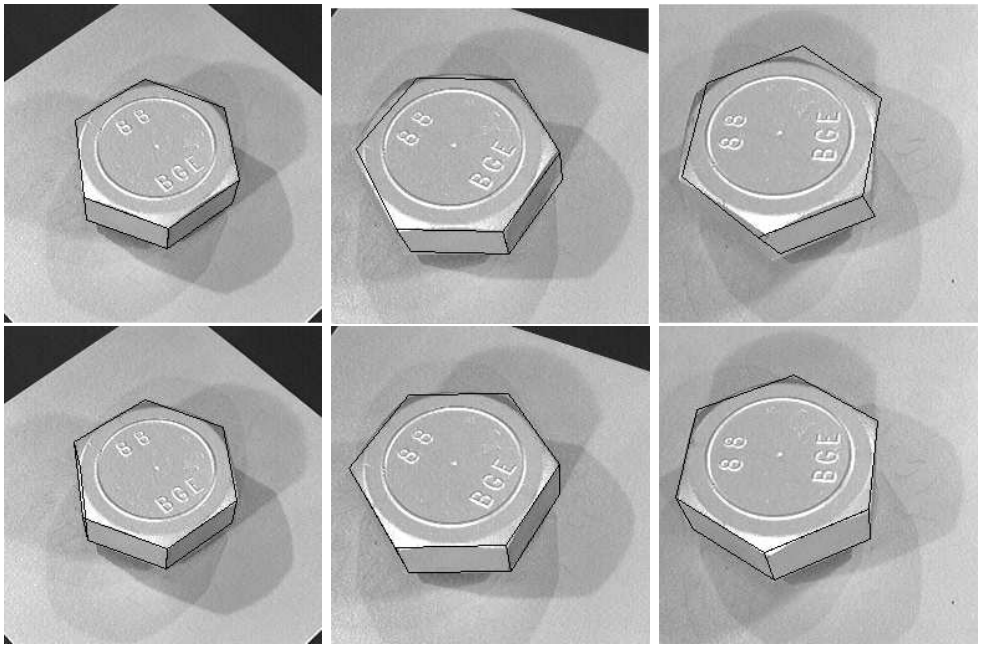

Figure 7: Nut tracking (images 256×256, $J = \pm 5$, $\lambda = 1500$): (a) tracking with only the 2D stage, (b) tracking with both 2D motion estimation and 3D pose computation stage

is performed around the y axis[1]. A face of the nut appears while another disappears. In Fig. 8.b, the main difficulty is the very important rotation around the x axis. Furthermore, the illumination conditions are not constant along the sequence. In Fig. 8.c, partial occlusions of the nut occur. In Fig. 14.d, the nut is tracked within a highly textured environment during a visual servoing experiment (see subsection 4.3). In all these experiments, satisfactory results are obtained.

Tracking a serial connector We have evaluated our tracking method on a still more complex object. In the experiments reported in Fig. 9, we consider a serial port connector placed on a newspaper forming a "cluttered" background. We only built a rough approximate model. Here, we have also to deal with low intensity gradient images, specularities, and no precisely defined contours. The serial connector is successfully tracked over a 170 frames image sequence. The camera performs a large displacement around the object. A face of the object appears while another disappears. Tracking is performed at 3 Hz (this lower processing rate is mainly due to the size of the CAD model of the object comprising more contours and leading to consider a greater number of points ς in the minimization of the cost function).

On-line estimation of radial distortion We have also tried to estimate on-line the radial lens distortion. We have considered a simple object (a box) and a camera with an important, but unknown, radial distortion (see Fig. 10). The focal lens of the camera is 3.5mm. We estimate on-line the distortion with initial value set to 0. The estimation of the radial distortion improved when the object projection moves toward the image border since a better estimation of this parameter is then required. In that case, the tracking is performed at 1 Hz.

4.3 Tracking within visual servoing experiments

Visual servoing overview. Image-based visual servoing consists in specifying a task as the regulation in the image of a set of visual features $\mathbf{P}$ that have

[1] z axis follows the optical axis, while x axis is parallel to the image rows and y axis is parallel to image columns.

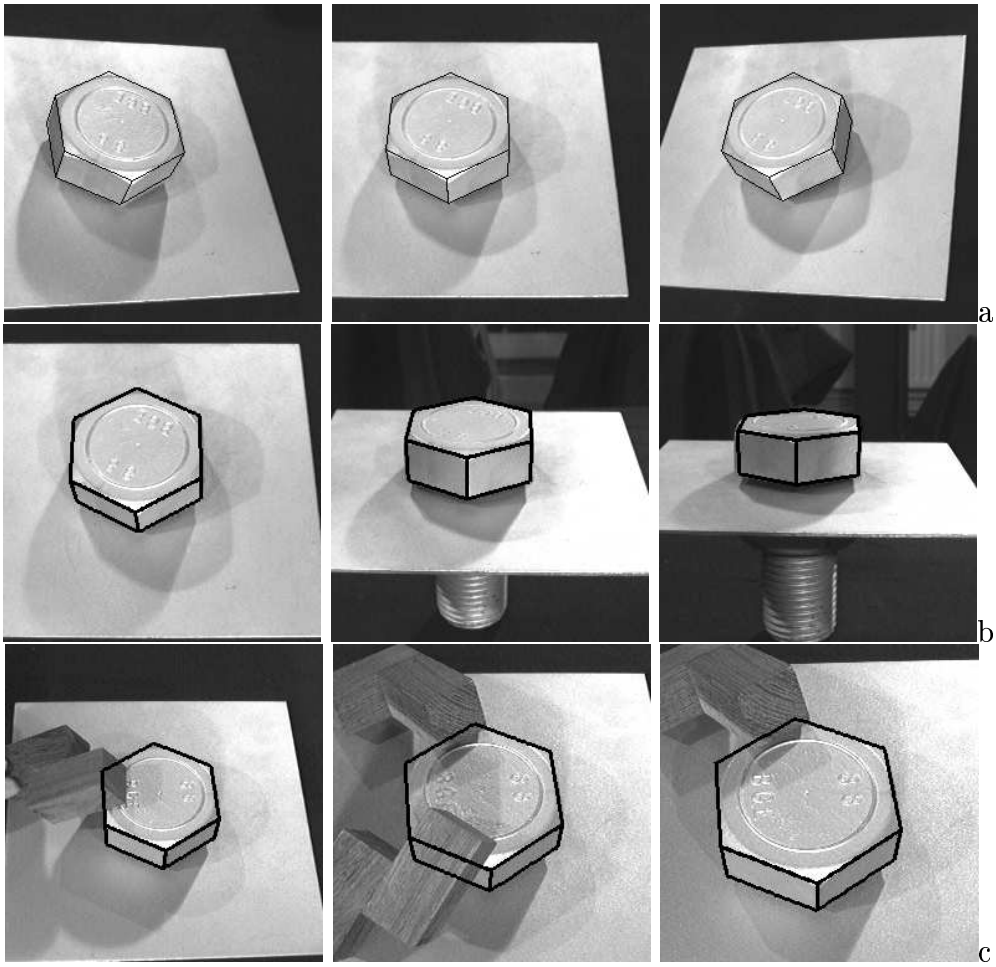

Figure 8: Successful nut tracking experiments featuring various difficulties (images 256×256, see text for details). $J = \pm 5$, $\lambda = 1500$

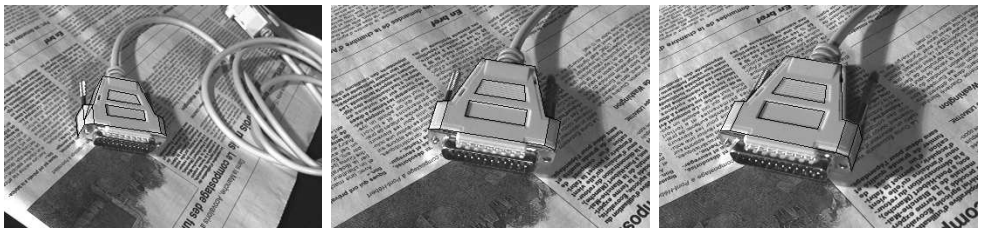

Figure 9: Tracking a connector on a newspaper background (images 365×256, $J = \pm 2$, $\lambda = 1500$.

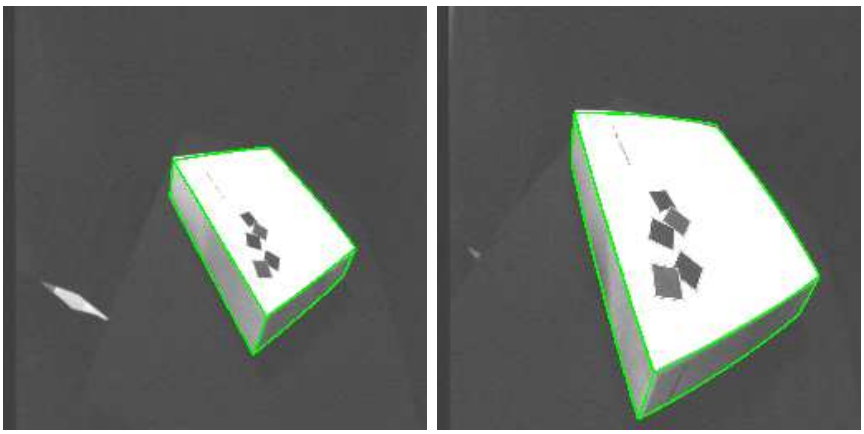

Figure 10: Handling distortion: distortion is very important in this example due to the use of a 3.5mm lens (images 512×512, $J = \pm 10, \lambda = 1500$).

to match a desired value $\mathbf{P_d}$ [9, 13]. The control law supplying an exponential decrease of the error $\mathbf{P} - \mathbf{P_d}$ in the image is given by [9]:

$$\mathbf{T_c} = -\lambda \mathbf{J}^+ (\mathbf{P} - \mathbf{P_d}) \tag{14}$$

where $\mathbf{T_c}$ is the camera velocity, λ is a scalar, and $\mathbf{J}^+$ denotes the pseudo inverse of the interaction matrix or image Jacobian $\mathbf{J}$ that links the camera motion to the image object motion.

Figure 11: Experimental setup at IRISA-INRIA Rennes: six degrees of freedom cartesian robot with a camera mounted on the end-effector.

We have considered positioning tasks. From an initial position of the robot, we want to reach a desired position expressed as a desired position of the object in the image. The complete implementation of the visual servoing task, including tracking and control, has been carried out on an experimental testbed involving a CCD camera mounted on the end effector of a six d.o.f cartesian robot (see Fig. 11). The realization of such an experiment involves the following steps:

- The camera is first positioned at the final desired position in order to learn of the targeted image features $\mathbf{P_d}$;

- The camera is then placed at the initial position. After a semi-automatic initialization as described in Section 4.1, current values of selected of the visual features $\mathbf{P}$ are computed from the results of the tracking algorithm at each iteration of the control low by backprojection of the CAD model knowing the pose. Camera motion is computed according to equation (14). The synoptic of the full process is given in Fig. 12.

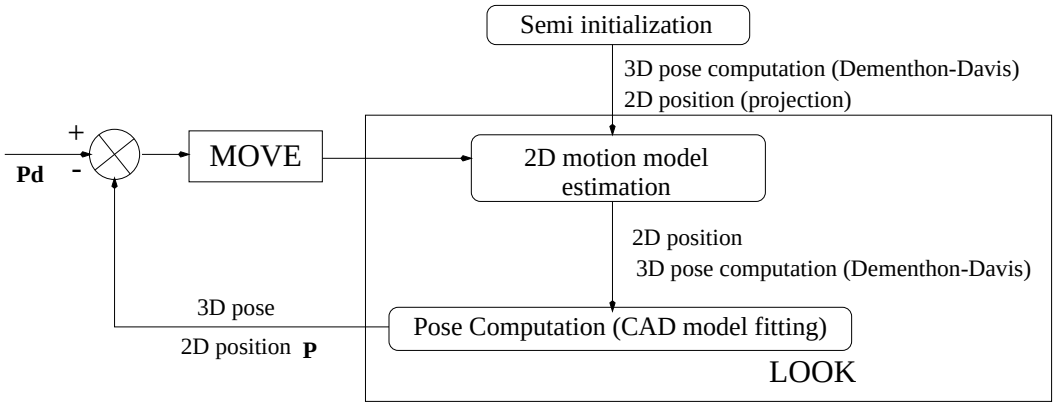

Figure 12: Overview of the visual servoing task.

Let us note that, even if we recover the object pose, i.e., the 3D position of the object wrt. the camera, we do not use this information within the visual servoing loop. Hence, we can cope with a rough camera calibration: the pose may be instable, or even biased, but this does not matter as long as the projection of the object model in the image is correct wrt. to the specification and the realization of the task at hand. 3D visual servoing [26] could also be considered but, in that case, instability and errors in pose computation may become a major problem.

Positioning with respect to a nut. Fig. 13a displays some of the images delivered by the camera during the positioning task. The current polygonal object model contours (in blue), depicting the results of the tracking of the nut projection in the image, and the desired final pattern (in red) are drawn over the images. In this experiment image features $\mathbf{P}$ are the six points of the

upper face of the nut. Fig. 13b shows the apparent trajectory in the image plane of points $\mathbf{P}$ during the achievement of the task. Fig. 13.c presents the temporal evolution of the error $\mathbf{P} - \mathbf{P_d}$. This demonstrates the stability and the convergence of the control law. The error on each coordinate of the six points specifying the task rapidly tends to zero. Noise appearing in the plots is mainly due to the fact that image processing is performed only at 3 Hz.

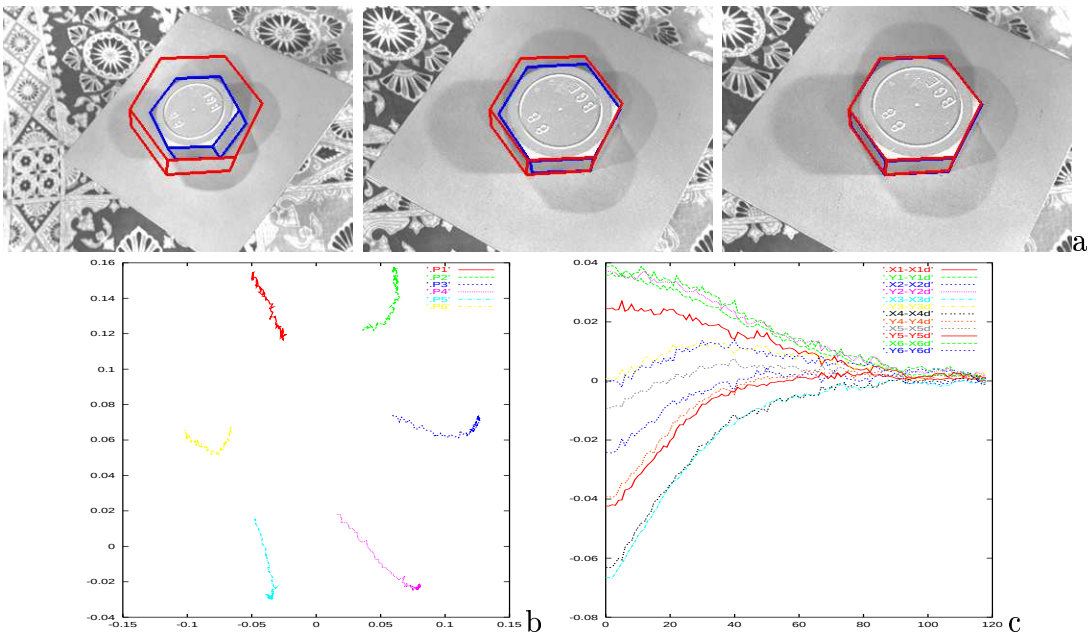

Figure 13: Positioning on the nut by visual servoing: (a) images supplied by the camera mounted the end-effector of the robot during the positioning task at time 15, 50 and 95. Red projection of the CAD model correspond to the desired position while the projection of tracked object appears in blue (image 365×256, $J = \pm 5$, $\lambda = 1500$) (b) temporal variation of the positions $\mathbf{P}$ of the control points in the image, (c) plots of the errors between the current and desired positions of the control points considered in the specification of the task.

Robustness. To prove the robustness of our algorithm we put the nut on a highly textured environment as shown in Fig. 14. As in the previous ex-

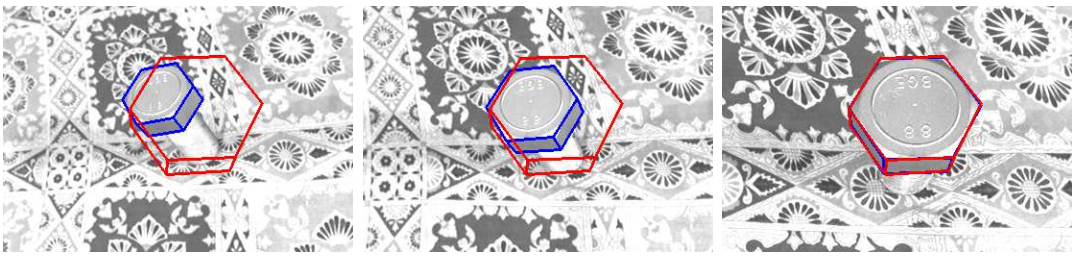

Figure 14: Positioning wrt. the nut by visual servoing within a highly textured environment at time 1, 40 and 154 (image 365×256, $J = \pm 3$, $\lambda = 1500$) .

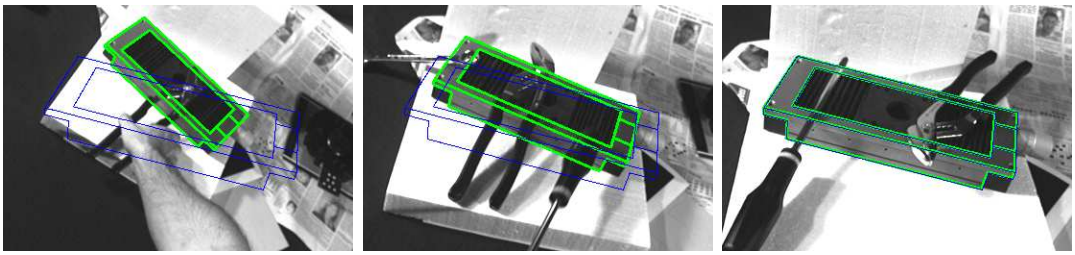

Figure 15: Positioning wrt. a micro-manipulation device (initial, middle and final position), the target pattern appears in blue (image 365×256, $J = \pm 5$, $\lambda = 1500$).

periments, our tracking algorithm embedded in the visual servoing scheme correctly achieves the positioning task wrt. the nut. Other experiments have been carried out using a micro-manipulation device as object of interest (see Fig. 15). Multiple temporary and partial occlusions by various tools have been imposed during the realization of the positioning task.

Accuracy. One of the application of a positioning task is grasping. In that context, repeatability is very important. The final position of the object in the image must be accurate enough and, in order to achieve the grasping task, the final 3D position of the robot end-effector has also to be consistent. The accuracy obtained in the positioning task wrt. the nut was computed from 40 experiments using the robot odometry (which is here very precise). We obtain an accuracy (expressed as the standard deviation of the final 3D position of the end-effector) of less that ± 0.7 *mm* in translation and ± 0.17 *deg* in rotation, while the object is located at 40 *cm* from the camera. The mean error $\mid \mathbf{P} - \mathbf{P_d} \mid$ is less than 0.2 pixels with a standard deviation of 0.3 pixels.

5 Conclusion

We have presented an original method for tracking complex objects in an image sequence at a high processing rate (but not yet exactly at video rate). The object tracking is a two step process based on the robust estimation between two successive images of a 2D global affine transformation undergone by the object projection, and on the computation of the object pose formulated as a cost minimization process. To perform this last step, an approximate polyhedral model of the object is sufficient. Appearance and disappearance of hidden faces of the object can be straightforwardly handled. Both steps of the tracking algorithm are robust to partial occlusions. The direct extension to non polyhedral object can be considered provided a 3D description of the object is available since this approach only requires to be able to project in the image the contours of the 3D object shape. This tracking algorithm allows us to efficiently realize visual servoing tasks in the robotics domain. We experimentally demonstrate this through various positioning tasks with respect to different real objects (without any landmarks) in complex situations. Visual

servoing is not the only application of this tracking method. Indeed, if we are able to achieve such a 2D tracking, we can also recover an appropriate precise estimation of the position of the camera wrt. the object if the camera is well calibrated, and then we can also perform a real 3D tracking.

Acknowledgments. This study has been supported by EdF - Pôle Industrie - Division Recherche et Développement under contract 1.97.C234. The image sequence of Fig. 10 has been supplied by LASMEA, University of Clermont-Ferrand, France.

Remark. Full results sequences can be found at `http://www.irisa.fr/vista` follows the *demonstrations* link then the *Robust real-time tracking* link.

References

[1] B. Bascle, P. Bouthemy, N. Deriche, and F. Meyer. Tracking complex primitives in an image sequence. In *Proc of Int. Conf. on Pattern Recognition, ICPR'94*, pages 426–431, Jerusalem, October 1994.

[2] M.-O. Berger. How to track efficiently piecewise curved contours with a view to reconstructing 3d objects. In *Proc of Int. Conf on Pattern Recognition, ICPR'94*, pages 32–36, Jerusalem, October 1994.

[3] S. Boukir, P. Bouthemy, F. Chaumette, and D. Juvin. A local method for contour matching and its parallel implementation. *Machine Vision and Application*, 10(5/6):321–330, April 1998.

[4] P. Bouthemy. A maximum likelihood framework for determining moving edges. *IEEE Trans. on Pattern Analysis and Machine intelligence*, 11(5):499–511, May 1989.

[5] P. Braud, M. Dhome, J.-T. Lapresté, and B. Peuchot. Reconnaissance, localisation et suivi d'objets polyhédriques par vision multi-oculaire. *Technique et Science Informatiques*, 16(1):9–38, Janvier 1997.

[6] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Active shape models - their training and application. *CVGIP : Image Understanding*, 61(1):38–59, Janvier 1994.

[7] N. Daucher, M. Dhome, J.T. Lapreste, and G. Rives. Modelled object pose estimation and tracking by monocular vision. In *British Machine Vision Conference, BMVC'93*, pages 249–258, Guildford, UK, September 1993.

[8] D. Dementhon and L. Davis. Model-based object pose in 25 lines of codes. *Int. J. of Computer Vision*, 15:123–141, 1995.

[9] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Trans. on Robotics and Automation*, 8(3):313–326, June 1992.

[10] D.B. Gennery. Visual tracking of known three-dimensional objects. *Int. J. of Computer Vision*, 7(3):243–270, 1992.

[11] N. Giordana, P. Bouthemy, F. Chaumette, F. Spindler, J.-C. Bordas, and V. Just. 2d model-based tracking of complex shapes for visual servoing tasks. in *Robust Vision for Vision-based Control of Motion*, M. Vincze, G. Hager (eds.), Chapitre 6, pages 67–75, IEEE Press, 2000.

[12] G. Hager and K. Toyama. The XVision system: A general-purpose substrate for portable real-time vision applications. *Computer Vision and Image Understanding*, 69(1):23–37, January 1998.

[13] S. Hutchinson, G. Hager, and P. Corke. A tutorial on visual servo control. *IEEE Trans. on Robotics and Automation*, 12(5):651–670, October 1996.

[14] M. Isard and A. Blake. Contour tracking by stochastic propagation of conditional density. In *European Conference on Computer Vision, ECCV'96,* Cambridge, UK, LNCS vol. 1064, Springer-Verlag, pages 343–356, 1996.

[15] J.K. Jain and A.K. Jain. Displacement measurement and its application in interframe image coding. *IEEE Trans. on Communications*, 29(12):1799–1808, December 1981.

[16] M. Kass, A. Witkin, and D. Terzopolous. Snakes : Active contour models. In *Proc. of Int. Conf. Computer Vision, ICCV'87*, pages 259–268, London, UK, 1987.

[17] C. Kervrann and F. Heitz. A hierarchical Markov modeling approach for the segmentation and tracking of deformable shapes. *Graphical Models and Image Processing*, 60(3):173–195, May 1998.

[18] H. Kollnig and H.-H. Nagel. 3D pose estimation by fitting image gradients directly to polyhedral models. In *IEEE Int. Conf. on Computer Vision*, pages 569–574, Boston, May 1995.

[19] D.G. Lowe. Robust model-based motion tracking through the integration of search and estimation. *Int. J. of Computer Vision*, 8(2):113–122, 1992.

[20] E. Marchand. Visp: A software environment for eye-in-hand visual servoing. In *IEEE Int. Conf. on Robotics and Automation, ICRA'99*, volume 4, pages 3224–3229, Detroit, Michigan, May 1999.

[21] E. Marchand, P. Bouthemy, F. Chaumette, and V. Moreau. Robust real-time visual tracking using a 2D-3D model-based approach. In *IEEE Int. Conf. on Computer Vision, ICCV'99*, volume 1, pages 262–268, Corfou, Grêce, September 1999.

[22] J.-M. Odobez and P. Bouthemy. Robust multiresolution estimation of parametric motion models. *Journal of Visual Communication and Image Representation*, 6(4):348–365, December 1995.

[23] J.-M. Odobez, P. Bouthemy, and E. Fleuet. Suivi 2D de pièces métalliques en vue d'un asservissement visuel. In *11th congrès RFIA'98*, volume 2, pages 173–182, Clermont-Ferrand, January 1998.

[24] A. Pece and A. Worrall. A statistically-based Newton method for pose refinement. *Image and Vision Computing*, 16(8):541–544, 1998.

[25] M. Tonko, K. Schäfer, F. Heimes, and H.-H. Nagel. Towards visually servoed manipulation of car engine parts. In *Proc. IEEE Int. Conf. on Robotics and Automation, ICRA'97*, volume 4, pages 3166–3171, Albuquerque, April 1997.

[26] W. Wilson, C. Hulls, and G. Bell. Relative end-effector control using cartesian position-based visual servoing. *IEEE Trans. on Robotics and Automation*, 12(5):684–696, October 1996.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399